
# Mininet-WPAN: Enabling WPAN and IoT Emulations with IEEE 802.15.4 Support

**Cassius Filho**[1], **Tiago Souza**[1], **Ramon Fontes**[1], **Eduardo Cerqueira**[2]

[1]Leading Advanced Technologies Center of Excellence (LANCE)
Universidade Federal do Rio Grande do Norte (UFRN) - RN - Brasil

[2]Instituto de Tecnologia
Universidade Federal do Pará (UFPA) - PA - Brasil

{cassius.filho.700, tiago.souza.100@}@ufrn.edu.br,
ramon.fontes@ufrn.br, cerqueira@ufpa.br

***Abstract.*** *The rapid growth of the Internet of Things (IoT) has created a significant demand for reliable and scalable emulation platforms capable of supporting low-power, wireless networks. In this context, IEEE 802.15.4 has emerged as a key protocol for enabling Wireless Personal Area Networks (WPANs) in IoT applications. This paper introduces Mininet-WPAN, an extension of the Mininet-WiFi emulator designed to support IEEE 802.15.4-based networks through the integration of the mac802154 Linux wireless device driver. Mininet-WPAN enables emulation of WPANs, allowing researchers and developers to explore and optimize various network configurations and protocols, including the Routing Protocol for Low-Power and Lossy Networks (RPL). We present three use cases demonstrating Mininet-WPAN's ability to emulate RPL in both storing and non-storing modes, showcasing its flexibility for IoT research. By offering lightweight virtualization, Software-Defined Networking (SDN) integration, and customizable network interfaces, Mininet-WPAN provides a powerful tool for advancing the development of IoT networks. This work also contributes to the open-source community by making the platform freely available for further research and innovation in low-power wireless communication.*

## 1. Introduction

The Internet of Things (IoT) has revolutionized the way we interact with the world by connecting a wide range of devices and enabling them to communicate and share data [Li et al. 2015]. In this context, the IEEE 802.15.4 protocol [IEEE 2015] serves as the foundation for many low-power and low-rate wireless personal area networks (WPANs), making it a critical component of the Internet of Things (IoT). As IoT devices continue to proliferate across various applications, from smart homes to industrial automation, the demand for reliable and efficient communication protocols has never been greater. In this scenario, network emulation plays a vital role by providing a controlled environment to simulate the unique characteristics and constraints of IEEE 802.15.4 networks, allowing researchers and developers to validate their designs before deployment.

One of the key challenges associated with IEEE 802.15.4 networks is the need to accommodate the inherent limitations of low-power devices, which often operate under resource-constrained conditions [Adeel et al. 2019]. These devices typically rely on

battery power and have limited processing capabilities, necessitating efficient communication mechanisms. Once again, network emulation enables the exploration of various configurations and scenarios, helping to identify optimal parameters for channel access, data transmission rates, and power management strategies. This process ensures that the protocols developed for IEEE 802.15.4 networks are robust and capable of functioning effectively under real-world conditions.

In addition to accommodating device constraints, network emulation also facilitates the study of interference and signal degradation, which are common in wireless environments. The IEEE 802.15.4 protocol operates in the crowded 2.4 GHz frequency band, where it may be affected by various sources of interference, including Wi-Fi, microwave ovens, and other wireless devices [Hassan et al. 2017]. By simulating these interference patterns in a controlled environment, researchers can evaluate the performance of IEEE 802.15.4 networks and implement strategies to mitigate the impact of external factors. This capability is crucial for enhancing the reliability and resilience of IoT applications relying on IEEE 802.15.4.

Network emulation supports the testing and validation of advanced features of the IEEE 802.15.4 protocol, such as mesh networking and adaptive frequency agility. These features are essential for improving the scalability and performance of IoT networks, particularly in challenging deployment scenarios. By allowing developers to experiment with different network topologies and configurations, network emulation enables the fine-tuning of protocols to ensure they meet the specific requirements of various applications. To facilitate this process, this paper introduces Mininet-WPAN, an extension of Mininet-WiFi [Fontes et al. 2015] designed to support wireless personal area networks (WPANs) by incorporating virtualized WPAN sensors/IoT devices based on the mac802154 Linux wireless device driver. As a proof of concept for initial experiments with Mininet-WPAN, this paper also presents three use cases: (i) without routing protocol; (ii) with routing protocol; and (iii) about energy consumption.

The remainder of this paper is organized as follows. Section 2 outlines the motivations behind this work; Section 3 describes related works; Section 4 describes the system architecture of Mininet-WPAN. Section 5 describes the case studies; and, finally, Section 6 concludes the paper.

## 2. Motivation

The motivation behind the development of Mininet-WPAN stems from the increasing demand for flexible, low-cost, and reproducible environments to study and experiment with wireless communication technologies, especially in the context of the Internet of Things (IoT). As wireless networks become the backbone of many IoT applications—from smart cities to industrial automation—there is a growing need for tools that allow researchers and developers to test and validate new ideas without relying on costly and complex physical testbeds.

Traditional network emulators, such as Mininet and Mininet-WiFi, have proven to be extremely valuable in the context of wired and wireless networks. However, there has been a noticeable gap when it comes to emulating wireless personal area networks (WPANs), particularly those based on standards like IEEE 802.15.4. This gap limits the ability to experiment with protocols specifically designed for low-power and lossy

networks, such as 6LoWPAN, RPL, and CoAP, which are fundamental to modern IoT ecosystems.

Mininet-WPAN addresses this gap by providing an extensible and lightweight environment capable of emulating wireless communications and network topologies in software. This makes it possible to perform rapid prototyping, debug protocol implementations, and analyze system performance under controlled and reproducible conditions. Moreover, by lowering the barrier to entry, it empowers students, educators, and researchers to engage with wireless network concepts without the need for expensive hardware setups.

In addition to its research and development benefits, Mininet-WPAN also contributes to the open-source community, fostering collaboration and accelerating innovation in the field. It facilitates the exploration of real-world wireless phenomena such as interference, mobility, and energy constraints, offering insights that are essential for the design of efficient and resilient wireless systems.

Thus, the creation of Mininet-WPAN is not only a technical contribution, but a response to a clear and growing need in the networking and IoT research communities: the ability to experiment, validate, and innovate in wireless communication—quickly, affordably, and at scale.

## 3. Related Works

When comparing Mininet-WPAN with other related emulation platforms, few tools stand out for their contributions to the field of network experimentation, particularly for low-power wireless networks, such as IEEE 802.15.4. Notable examples (see Table 1) include Cooja (a Contiki OS-based simulator) [OS 2024], ns-3 [ns 3 2024], and OMNeT++ [OMNeT++ 2024]. Each of these tools offers unique capabilities, but they also come with limitations compared to Mininet-WPAN.

**Table 1. Overview of Emulation Software**

| Software | Last Activity | Type | Network | Language |
|---|---|---|---|---|
| ns-3 | 2024 | Sim | Any | C, Python |
| OMNeT++ | 2024 | Sim | Any | C++ |
| Cooja | 2018 | Sim/Emu | WSN | C (TinyOS) |

Ns-3 offers powerful network simulation capabilities for both wireless and wired networks. It has a comprehensive implementation of the IEEE 802.15.4 MAC and PHY layers, allowing detailed analysis of low-power networks. However, ns-3 is primarily a discrete-event simulator, which can limit its real-time network interaction capabilities. Furthermore, ns-3 uses API-specific glue code for its POSIX and kernel support, which does not cover system calls for all applications, making it less suitable for scenarios requiring fine-grained control of network components or SDN-based experimentation.

Cooja is an obsolete simulation tool integrated with Contiki OS, mainly used for simulating large-scale IoT networks. It provides detailed node-level simulations and supports a range of low-power protocols such as RPL (Routing Protocol for Low-Power and Lossy Networks). While Cooja excels at simulating large-scale networks and offers seam-

less integration with Contiki's IoT protocols, its focus on detailed sensor network simulations can lead to higher complexity and slower execution when scaling to larger scenarios. Additionally, it lacks the flexibility to integrate with software-defined networking (SDN) environments, which Mininet-WPAN supports through Mininet's SDN features.

OMNeT++ is another popular network simulation tool with broad support for IoT protocols, including IEEE 802.15.4. OMNeT++ is highly modular and customizable, which makes it a preferred choice for large-scale, complex network simulations. However, it shares similar limitations to ns-3 in terms of real-time performance and the lack of SDN-native features, making it less optimal for scenarios that require virtualized, scalable, and easily configurable testbeds.

Mininet-WPAN leverages the strengths of the Mininet-WiFi architecture by extending it to support wireless personal area networks (WPANs), specifically focusing on the IEEE 802.15.4 protocol. It stands out from the tools mentioned above in several ways:

- **Real-time Emulation**: unlike traditional simulators like ns-3 and OMNeT++, Mininet-WPAN operates as an emulator, allowing real-time experimentation and interaction with actual network configurations. This provides a more realistic environment for testing;
- **SDN Integration**: built on top of Mininet, Mininet-WPAN inherits strong support for SDN. This enables users to experiment with software-defined networking technologies in low-power networks, a feature not natively available in other platforms like Cooja or ns-3;
- **Lightweight Virtualization**: Mininet-WPAN supports lightweight virtualization, making it scalable and efficient for testing large networks without the overhead of full virtual machine simulations. This makes it particularly suitable for rapid experimentation, debugging, and prototyping;
- **Customization and Flexibility**: the use of the mac802154 Linux wireless device driver allows Mininet-WPAN to support flexible, low-level manipulation of network interfaces. It is ideal for researchers who need fine-grained control over network behaviors, which is harder to achieve in the more abstract environments of Cooja or OMNeT++.

## 4. Mininet-WPAN

Figure 1 illustrates the architecture and components involved in the interaction between Mininet-WPAN[1], Linux kernel components, and IEEE 802.15.4-related tools and subsystems. Mininet-WPAN emulates WPAN using IEEE 802.15.4 interfaces. The framework creates virtual WPAN interfaces (wpan0) for each emulated sensor node (sensor1-wpan0, sensor2-wpan0, etc.) inside isolated namespaces (`/bin/bash`) - *docker containers are also supported by means of containernet [Peuster et al. 2016].* These namespaces provide a shell environment for running applications or emulations related to wireless sensors.

Each namespace can run tools such as `iwpan` (for WPAN configuration), `ip` (for networking), and `RPL` (Routing Protocol for Low-power and Lossy Networks), among others. These tools can interact with the underlying network configuration and routing

[1]https://github.com/ramonfontes/containernet

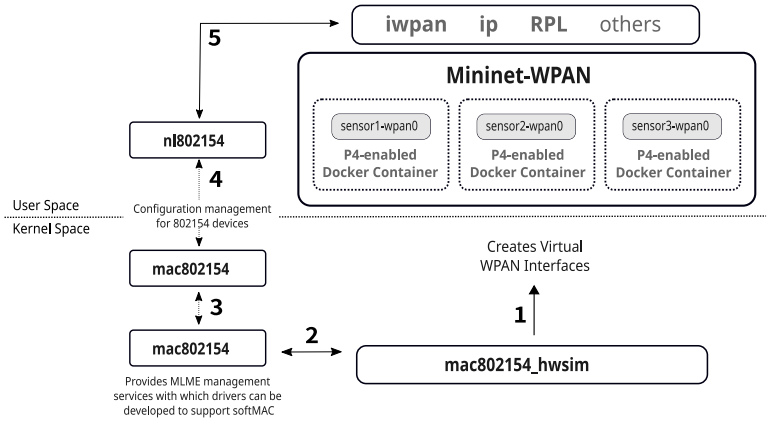


**Figure 1. Mininet-WPAN Architecture and Components.**

for WPAN nodes. In the userspace, the nl802154 component is responsible for managing IEEE 802.15.4 devices in Linux. It uses Netlink (nl) sockets to facilitate communication between userspace tools (e.g. iwpan) and kernel-space components. This interface allows for the configuration and management of WPAN devices, including tasks such as setting channels, PAN IDs, and other network parameters.

In kernel space, the mac802154 module implements the MAC (Medium Access Control) layer for IEEE 802.15.4 networks. It serves as the core component that manages and controls WPAN interfaces. The mac802154 module is responsible for handling key medium access control functions, including device addressing, data framing, and collision avoidance, which are necessary for communication between WPAN devices.

Additionally, the mac802154_hwsim module acts as a hardware simulator for IEEE 802.15.4 networks. This kernel module creates virtual WPAN interfaces, enabling Mininet-WPAN to emulate wireless network scenarios without the need for physical hardware. This simulation capability is essential for testing and development purposes, as it allows developers to emulate and analyze wireless sensor networks and their behavior in various conditions.

### 4.1. Communication Flow

The communication flow in Mininet-WPAN operates as follows:

1. Mininet-WPAN creates multiple virtual WPAN interfaces (e.g., sensor1-wpan0, sensor2-wpan0) within isolated sensor namespaces, enabling the simulation of sensor nodes;
2. These virtual interfaces are created by the mac802154_hwsim module, which simulates the underlying 802.15.4 hardware;
3. The mac802154 kernel module manages these interfaces, providing the core functionalities needed for wireless communication;
4. nl802154 acts as the communication bridge between the kernel's mac802154 and user-space tools (e.g. iwpan), facilitating the configuration and control of the simulated WPAN interfaces;
5. In userspace, tools like iwpan configure WPAN settings, ip handles IP networking, and protocols like RPL can be used to simulate network routing for IoT and low-power networks.

### 4.2. IPv6 Routing

In Mininet-WPAN, routing plays a pivotal role, especially in IoT scenarios where nodes often have constraints on power, memory, and processing capacity. To address these constraints, Mininet-WPAN incorporates RPL (Routing Protocol for Low-Power and Lossy Networks) [Thubert and Richardson 2021], a protocol designed for environments with unreliable communication links and resource-limited devices, which suits IEEE 802.15.4-based networks.

RPL organizes the network into a Destination-Oriented Directed Acyclic Graph (DoDAG), with nodes hierarchically structured, rooted at the top and branching out in levels. This structured graph allows efficient routing management that factors in energy consumption, link quality, and latency. RPL's flexibility comes from its use of various Objective Functions (OFs), enabling optimization for different network needs, making it suitable for scalable, diverse applications. To facilitate network operations, RPL employs the following control messages.

- **D**odag **I**nformation **S**olicitation (DIS): Sent by a node to request details about the DODAG (Destination-Oriented Directed Acyclic Graph) from potential parent nodes. DIS is primarily used by nodes during initialization or when they lack a known parent to discover routers and join an existing DODAG;
- **D**odag **I**nformation **O**bject (DIO): Contains information about the DODAG and is used to advertise the network topology. DIOs are periodically sent by root and intermediate nodes to inform others of the DODAG's structure, ID, level, and routing preferences;
- **D**estination **A**dvertisement **O**bject (DAO): Used by nodes to advertise available routes to specific destinations within the network. DAO updates routing tables and informs other nodes of reachable destinations, facilitating route management;
- **D**estination **A**dvertisement **O**bject **Ack**nowledgment (DAO-ACK): Sent in response to a DAO, confirming receipt and processing of the routing information, ensuring reliable route advertisement within the DODAG.

### 4.3. Limitations

Mininet-WPAN is an effective platform for emulating IEEE 802.15.4 networks and IoT scenarios but has key limitations. One major drawback is the lack of support for wmediumd, which is essential for simulating realistic wireless signal propagation, interference, and dynamic link quality. Without wmediumd [Fontes 2024], Mininet-WPAN struggles to accurately model physical layer behaviors, especially node mobility, impacting its ability to emulate changes in signal strength and link quality due to movement. This reduces the accuracy of mobility-related experiments, limiting insights into network performance factors such as handover, link breakage, and transmission quality variations.

Scalability will be certainly another challenge - although we have not performed scalability tests yet we hope to be able to emulate a few hundred of nodes; as network complexity increases with more nodes or intricate topologies, the platform's performance can suffer, affecting its capacity to emulate large-scale IoT scenarios. However, Mininet-WPAN remains suitable for many IoT applications that typically involve a limited number of devices, allowing it to support a broad range of use cases despite these limitations.

## 5. Use Cases

Three use cases will now be presented to showcase the capabilities of Mininet-WPAN: a scenario without RPL; other utilizing RPL and the last one about energy consumption. In these case studies, ten IEEE 802.15.4-based wireless sensors were emulated on an Ubuntu system running the 6.5.0-1025 kernel and all related artifacts are publicly available[2] to ensure reproducibility.

### 5.1. Use Case #1: Without the Routing Protocol

The goal of this use case is to demonstrate that it is possible to emulate IEEE 802.15.4 networks using the Linux Kernel stack, enabling communication among virtual sensors. This emulation setup offers a clear view of the architecture of IEEE 802.15.4 and 6LoWPAN (IPv6 over Low-power Wireless Personal Area Networks), where two specific interfaces are created to reflect the structure and functionality of a real WPAN network:

- `wpanX`: This interface handles raw IEEE 802.15.4 communication, focusing on the MAC and link layers.
- `panX`: Built on top of wpanX, this interface provides IPv6 support over IEEE 802.15.4, essential for IoT applications using 6LoWPAN.

This dual-interface design simplifies the development, testing, and implementation of IoT networks in a simulated environment, removing the need for physical hardware. It allows greater flexibility in configuration, experimentation, and understanding of the behavior of WPAN networks with IPv6 capabilities, making it an effective approach for IoT research and prototyping.

In addition to demonstrating the emulation of an IEEE 802.15.4 network, this use case also shows how to establish communication between two virtual sensors using the `ping` command. This setup not only validates the feasibility of creating an IEEE 802.15.4 network using the Linux Kernel stack but also illustrates practical sensor-to-sensor communication, allowing us to test network connectivity and latency in a controlled, emulated environment, as illustrated in Annex A.

### 5.2. Use Case #2: With the Routing Protocol

We take Mininet-WPAN's capabilities a step further in this use case by enabling the RPL protocol. The goal here is to demonstrate the feasibility of integrating a widely recognized RPL implementation, showcasing how Mininet-WPAN can support standardized IoT routing protocols. This example highlights Mininet-WPAN's flexibility and compatibility with established IoT networking standards, allowing users to create more complex and realistic network scenarios for low-power, multi-hop communication.

More specifically, we replicate the same network topology as before, but now the nodes communicate using the RPL routing protocol. This setup enables multi-hop communication across nodes, demonstrating how Mininet-WPAN supports routing in low-power, lossy network environments. By incorporating RPL, we illustrate how nodes can dynamically form and maintain routing paths, making this use case a practical example of scalable IoT networking in Mininet-WPAN.

[2]https://github.com/ramonfontes/mn-wpan

Using traceroute, Annex B reveal the sequence of nodes through which the packets traverse, with `sensor2` appearing as a hop between `sensor7` and `sensor1`. This behavior highlights the importance of intermediate nodes in enabling long-distance communication in networks where direct connections are not feasible due to range limitations.

### 5.3. Use Case #3: Energy Consumption

We now demonstrate how to gather information on battery consumption within the network. By monitoring energy usage, we gain insights into the power efficiency of each node, which is crucial for evaluating network performance in low-power IoT applications. This example highlights Mininet-WPAN's ability to emulate realistic sensor network constraints, helping users analyze and optimize energy consumption for extended network lifetime in battery-operated devices.

In this use case, each sensor operates at a defined voltage, and the energy consumption is calculated using the formula $P = V * I$, where the voltage is customized for each sensor and the current is determined based on the reference value defined in [Halperin et al. 2010]. This reference value depends on the state of the network interface (e.g., idle, transmitting), resulting in the power consumption in watts. The goal of this use case is to demonstrate that increased data transmission by a sensor directly correlates with higher energy consumption.

As observed though Annex C, `sensor1` and `sensor2` exhibit higher battery consumption due to the increased packet exchange between them. This interaction demonstrates the impact of communication load on energy usage, highlighting the need for efficient power management in networks with limited battery resources. By monitoring this battery consumption, we can gain valuable insights into the energy demands of active sensor nodes, which is essential for optimizing network lifetime in IoT applications. Mininet-WPAN can also be used to measure battery consumption based on CPU usage, as illustrated in Listing 4. This feature is particularly valuable, as it allows nodes to run in Docker containers, enhancing fidelity in capturing the specific battery consumption of each node.

## 6. Conclusion

This paper introduced Mininet-WPAN, an innovative extension of Mininet-WiFi designed to emulate IEEE 802.15.4-based networks, establishing a robust platform for IoT research and development. Leveraging real-time emulation and the mac802154 wireless driver, Mininet-WPAN provides a flexible, scalable environment for simulating WPANs. Case studies demonstrated the platform's capability to handle RPL protocol operations in both storing and non-storing modes, showcasing its adaptability to various network topologies and configurations.

By incorporating lightweight virtualization and SDN integration, Mininet-WPAN provides fine-grained network control while ensuring scalability and efficiency, making it well-suited for IoT applications requiring real-time emulation and advanced protocol testing. Future enhancements will focus on integrating wmediumd to simulate realistic wireless conditions, such as signal propagation and interference, improving the platform's ability to model mobile IoT devices and the effects of movement on network performance, including handovers and link quality variations.

## Acknowledgment

We thank the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brazil (CAPES) grant #88887.954253/2024-00 for financial support and Bolsista CAPES/Brasil' for postdoctoral grant (Process 88887.005666/2024-00)

## References

Adeel, A., Gogate, M., Farooq, S., Ieracitano, C., Dashtipour, K., Larijani, H., and Hussain, A. (2019). A survey on the role of wireless sensor networks and iot in disaster management. *Geological disaster monitoring based on sensor networks*, pages 57–66.

Fontes, R. (2024). wmediumd: Wireless medium model for network emulation. Accessed: 2024-10-21.

Fontes, R. R., Afzal, S., Brito, S. H., Santos, M. A., and Rothenberg, C. E. (2015). Mininet-wifi: Emulating software-defined wireless networks. In *2015 11th International conference on network and service management (CNSM)*, pages 384–389. IEEE.

Halperin, D., Greenstein, B., Sheth, A., and Wetherall, D. (2010). Demystifying 802.11 n power consumption. In *Proceedings of the 2010 international conference on Power aware computing and systems*, page 1. USENIX Association Berkeley, CA, USA.

Hassan, R., Rahman, M., and Shakir, M. (2017). Performance analysis of ieee 802.15.4 for wireless sensor networks. *International Journal of Computer Applications*, 175(6):30–35.

IEEE (2015). Ieee standard for low-rate wireless personal area networks (lr-wpans). Accessed: 2024-10-21.

Li, S., Xu, L. D., and Zhao, S. (2015). The internet of things: a survey. *Information systems frontiers*, 17:243–259.

ns 3 (2024). ns-3: Discrete-event network simulator. Accessed: 2024-10-21.

OMNeT++ (2024). Omnet++: Discrete event simulator. Accessed: 2024-10-21.

OS, C. (2024). Contiki: The open source operating system for the internet of things. Accessed: 2024-10-21.

Peuster, M., Karl, H., and van Rossem, S. (2016). Medicine: Rapid prototyping of production-ready network services in multi-pop environments. In *2016 IEEE Conference on Network Function Virtualization and Software Defined Networks (NFV-SDN)*, pages 148–153.

Thubert, P. and Richardson, M. (2021). Routing for RPL (Routing Protocol for Low-Power and Lossy Networks) Leaves. RFC 9010.

## A. Annex A - Use Case #1

**Listing 1. Pinging sensor1 to sensor2**

```
> sensor1 ping -c1 fe80::2%sensor1-pan0
PING fe80::2%sensor1-pan0(fe80::2%sensor1-pan0) 56 data bytes
64 bytes from fe80::2%sensor1-pan0: icmp_seq=1 ttl=64 time=0.099 ms
```

--- fe80::2%sensor1-pan0 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 0.099/0.099/0.099/0.000 ms

## B. Annex B - Use Case #2

**Listing 2. Traceroute**

root@sensor1:/# traceroute fd3c:be8a:173f:8e80:3083:6218:aee4:e62d
traceroute to fd3c:be8a:173f:8e80:3083:6218:aee4:e62d, 30 hops max, 80 byte packets
 1 fd3c:be8a:173f:8e80:419:c082:e8cc:26a0 (fd3c:be8a:173f:8e80:419:c082:e8cc:26a0) 0.319 ms 0.047 ms 0.034 ms
 2 fd3c:be8a:173f:8e80:3083:6218:aee4:e62d (fd3c:be8a:173f:8e80:3083:6218:aee4:e62d) 0.109 ms 0.063 ms 0.068 ms

## C. Annex C - Use Case #3

**Listing 3. Energy Consumption Result by the Network Interface of the Sensors**

energy consumed by sensor1: 58.64869999999997 mW
energy consumed by sensor2: 58.64869999999997 mW
energy consumed by sensor7: 53.22450000000004 mW

**Listing 4. Energy Consumption Result by the CPU usage of the Sensors**

energy consumed by sensor1: 6.0833333333333335e-05 Wh
energy consumed by sensor2: 3.166666666666667e-05 Wh
energy consumed by sensor7: 2.7750000000000007e-05 Wh